

Online Natural Gradient as a Kalman Filter

Yann Ollivier

Abstract

We establish a full relationship between Kalman filtering and Amari’s natural gradient in statistical learning. Namely, using an online natural gradient descent on data log-likelihood to evaluate the parameter of a probabilistic model from a series of observations, is exactly equivalent to using an extended Kalman filter to estimate the parameter (assumed to have constant dynamics).

In the i.i.d. case, this relation is a consequence of the “information filter” phrasing of the extended Kalman filter. In the recurrent (state space, non-i.i.d.) case, we prove that the joint Kalman filter over states and parameters is a natural gradient on top of real-time recurrent learning (RTRL), a classical algorithm to train recurrent models.

This exact algebraic correspondence provides relevant settings for natural gradient hyperparameters such as learning rates or initialization and regularization of the Fisher information matrix.

Suppose we have a series of observation pairs $(u_1, y_1), \dots, (u_t, y_t), \dots$ and want to predict y_t from u_t using a probabilistic model p_θ . Assume that the model for y_t is a Gaussian centered on a predicted value \hat{y}_t , with known covariance matrix R_t , namely

$$y_t = \hat{y}_t + \mathcal{N}(0, R_t), \quad \hat{y}_t = h(\theta, u_t) \quad (1)$$

For instance, h may represent a feedforward neural network with input u , parameters θ , and output \hat{y} . The goal is to find the parameters θ such that the prediction $\hat{y}_t = h(\theta, u_t)$ is as close as possible to y_t : the loss function is

$$\ell_t = \frac{1}{2}(\hat{y}_t - y_t)^\top R_t^{-1}(\hat{y}_t - y_t) = -\ln p(y_t|\hat{y}_t) \quad (2)$$

up to an additive constant.

For non-Gaussian outputs, we assume that the noise model on y_t given \hat{y}_t belongs to an exponential family, namely, that \hat{y}_t is the mean parameter of an exponential family of distributions¹ over y_t ; we again define the loss function as $\ell_t := -\ln p(y_t|\hat{y}_t)$, and we define R_t as the covariance matrix of the sufficient statistics of y_t given this mean. For a Gaussian output noise

¹ The Appendix contains a reminder on exponential families. An *exponential family of probability distributions* on y , with sufficient statistics $T_1(y), \dots, T_K(y)$, and with

this works as expected. For instance, for a classification problem, the output is categorical, $y_t \in \{1, \dots, K\}$, and \hat{y}_t will be the set of probabilities $\hat{y}_t = (p_1, \dots, p_{K-1})$ to have $y_t = 1, \dots, K-1$. In that case R_t is the $(K-1) \times (K-1)$ matrix $(R_t)_{kk'} = \text{diag}(p_k) - p_k p_{k'}$. (The last probability p_K is determined by the others via $\sum p_k = 1$ and has to be excluded to obtain a non-degenerate parameterization and an invertible covariance matrix R_t .)

The Kalman filter extends very naturally to such a setting (Def. 7), just by replacing the measurement error $y_t - \hat{y}_t$ with $T(y_t) - \hat{y}_t$, with T the sufficient statistics for the exponential family. (For Gaussian noise this is the same, as $T(y)$ is y .)

First, we state and prove the exact equivalence between two algorithms for this kind of problem, namely, Amari’s *natural gradient descent* applied online [Ama98], and the *extended Kalman filter* applied to the parameter θ treated as a fixed hidden variable. The only assumption is that the output noise on y given the prediction \hat{y} is modelled by an exponential family (e.g., Gaussian or categorical); the prediction \hat{y} itself may result from any model, such as a feedforward neural network.

Next, the *recurrent* (or state space model) case, in which consecutive data pairs (u_t, y_t) are not independent and are treated with models such as recurrent neural networks, is treated in §6. We prove that the extended Kalman filter applied to the state and parameter, exactly amounts to a natural gradient on top of *real-time recurrent learning*, a classical (and costly) online algorithm for recurrent network training [Jae02].

Table 1 lists the main correspondences between objects from the Kalman filter side and from the natural gradient side, as results from the theorems and propositions below. Note that the correspondence is one-sided: the natural gradient is exactly an extended Kalman filter, but only those Kalman filters applied to parameter estimation problems (namely, with constant dynamics on the parameter part of the system) appear as a natural gradient.

We refer to [SW88] for an early example of the use of Kalman filtering for training feedforward neural networks, to [Jae02] for a review of techniques for the recurrent case (including Kalman filters) and to [Hay01] for a clear, in-depth treatment of Kalman filtering for recurrent models.

In the i.i.d. (non-recurrent) case, the correspondence follows from the parameter $\beta \in \mathbb{R}^K$, is given by

$$p_\beta(y) := \frac{1}{Z(\beta)} e^{\sum_k \beta_k T_k(y)} \lambda(dy) \quad (3)$$

where $Z(\beta)$ is a normalizing constant, and $\lambda(dy)$ is any reference measure on y . For instance, if $y \in \mathbb{R}^K$, $T_k(y) = y_k$ and $\lambda(dy)$ is a Gaussian measure centered on 0, by varying β one gets all Gaussian measures with the same covariance matrix and another mean. y may be discrete, e.g., Bernoulli distributions correspond to λ the uniform measure on $y \in \{0, 1\}$ and a single sufficient statistic $T(0) = 0$, $T(1) = 1$. Often, the *mean parameter* $\bar{T} := \mathbb{E}_{y \sim p_\beta} T(y)$ is a more convenient parameterization than β . Exponential families maximize entropy (minimize information divergence from λ) for a given mean of T .

<i>iid (non-recurrent) model $\hat{y}_t = h(\theta, u_t)$</i>	
Extended Kalman filter on θ	Online natural gradient on θ with learning rate $\eta_t = 1/(t+1)$
Covariance matrix P_t	Fisher information matrix $J_t = \eta_t P_t^{-1}$
Bayesian prior P_0	Fisher matrix initialization $J_0 = P_0^{-1}$
Fading memory	Larger learning rate
Fading memory+constant prior	Fisher matrix regularization
<i>Recurrent (state space) model $\hat{y}_t = \Phi(\hat{y}_{t-1}, \theta, u_t)$</i>	
Extended Kalman filter on (θ, \hat{y})	RTRL+natural gradient+state correction
Covariance of θ alone, P^θ	Fisher matrix $J_t = \eta_t (P^\theta)^{-1}$
Correlation between θ and \hat{y}_t	Gradient estimate $\frac{\partial \hat{y}_t}{\partial \theta}$

Table 1: Natural gradient objects vs Kalman filter objects.

information filter phrasing of Kalman filtering [Sim06, §6.2] by simple manipulations. Nevertheless, we could not find in the literature an explicit reference identifying the two. For instance, [Ber96] identifies the extended Kalman filter with a Gauss–Newton gradient descent for the specific case of nonlinear regression; [ŠKT01] uses the Fisher information matrix to study the variance of parameter estimation in Kalman-like filters, without using a natural gradient; [BL03] comment on the similarity between Kalman filtering and a version of Amari’s natural gradient, again in the specific case of least squares regression; [Pat16] exploits the relationship between second-order gradient descent and Kalman filtering in specific cases including linear regression; [LCL⁺17] use a natural gradient descent over Gaussian distributions for an auxiliary problem arising in Kalman-like Bayesian filtering, a problem independent from the one treated here. In the recurrent (non-i.i.d.) case, [Hay01, §5] contains statements that can be interpreted as relating Kalman filtering and preconditioned RTRL-like gradient descent for recurrent models (§6 below).

Casting the natural gradient as a specific case of the extended Kalman filter is an instance of the provocative statement from [LS83] that “there is only one recursive identification method” that is optimal on quadratic functions. Indeed, the online natural gradient descent fits into the framework of [LS83, §3.4.5]. Arguably, this statement is limited to linear models, and for non-linear models one should expect different algorithms to coincide only at a certain order; however, the correspondences presented below are exact.

None of these algorithms scales well to large-dimensional models as currently used in machine learning, so that approximations are required. The correspondence raises the possibility that various methods developed for Kalman filtering (e.g., particle or unscented filters) or for natural gradient approximations (e.g., matrix factorizations) could be transferred from one viewpoint to the other. Proof techniques could be transferred as well: for instance, Amari [Ama98] gave a strong but sometimes informal argument that the natural gradient is Fisher-efficient, i.e., the resulting parameter estimate

is asymptotically optimal for the Cramér–Rao bound; alternate proofs could be obtained by transferring related statements for the extended Kalman filter, e.g., combining techniques from [ŠKT01, BRD97, LS83].

Acknowledgments. Many thanks to Silvère Bonnabel and Gaétan Marceau-Caron for their careful reading of the manuscript, corrections, and suggestions for the organization of the text. I would also like to thank Shun-ichi Amari and Frédéric Barbaresco for additional comments and for pointing out relevant references.

Notation. In statistical learning, the external inputs or regressor variables are often denoted x . In Kalman filtering, x often denotes the state of the system, while the external inputs are often u . Thus we will avoid x altogether and denote by u the inputs and by s the state of the system.

The natural gradient descent on parameter θ_t will use the Fisher matrix J_t . The Kalman filter will have posterior covariance matrix P_t .

The variable to be predicted at time t will be y_t , and \hat{y}_t is the corresponding prediction. In general \hat{y}_t and y_t may be different objects in that \hat{y}_t encodes a full probabilistic prediction for y_t . For Gaussians with known variance, \hat{y}_t is just the predicted mean of y_t , so in this case y_t and \hat{y}_t are the same type of object. For Gaussians with unknown variance, \hat{y} encodes both the mean and second moment of y . For discrete categorical data, \hat{y} encodes the probability of each possible outcome y .

For multidimensional quantities x and $y = f(x)$, we denote by $\frac{\partial y}{\partial x}$ the Jacobian matrix of y w.r.t. x , whose (i, j) entry is $\frac{\partial f_i(x)}{\partial x_j}$. This satisfies the chain rule $\frac{\partial z}{\partial y} \frac{\partial y}{\partial x} = \frac{\partial z}{\partial x}$. With this convention, gradients of real-valued functions are *row* vectors, so that a gradient descent takes the form $x \leftarrow x - \eta (\partial f / \partial x)^\top$.

For a column vector u , $u^{\otimes 2}$ is synonymous with uu^\top , and with $u^\top u$ for a row vector.

§1 Natural gradient descent. A standard approach to optimize the parameter θ of a probabilistic model is an online gradient descent

$$\theta_t \leftarrow \theta_{t-1} - \eta_t \frac{\partial \ell_t(y_t)^\top}{\partial \theta} \quad (4)$$

with learning rate η_t . The *natural gradient* attempts to solve several practical and theoretical shortcomings of simple gradient descent, by preconditioning this gradient descent with $J(\theta)^{-1}$ where J is the *Fisher information matrix* with respect to the parameter θ (Definition 1 below). For instance, with the simple gradient descent, simple changes in parameter encoding or in data presentation (e.g., encoding black and white in images by 0/1 or 1/0) can result in different learning performance. The natural gradient achieves invariance with respect to parameter re-encoding; in particular, learning become insensitive to the characteristic scale of each parameter direction, so that different directions naturally get suitable learning rates.

However, this comes at a large computational cost for large-dimensional models: just storing the Fisher matrix already costs $O((\dim \theta)^2)$. Various

strategies are available to approximate the natural gradient for complex models such as neural networks, using diagonal or block-diagonal approximation schemes for the Fisher matrix, e.g., [LMB07, Oll15, MCO16, GS15, MG15].

DEFINITION 1 (ONLINE NATURAL GRADIENT). *Consider a statistical model with parameter θ that predicts an output y given an input u . Suppose that the prediction takes the form $y \sim p(y|\hat{y})$ where $\hat{y} = h(\theta, u)$ depends on the input via a model h with parameter θ . Given observation pairs (u_t, y_t) , the goal is to minimize, online, the loss function*

$$\sum_t \ell_t(y_t), \quad \ell_t(y) := -\ln p(y|\hat{y}_t) \quad (5)$$

as a function of θ .

The online natural gradient maintains a current estimate θ_t of the parameter θ , and a current approximation of the Fisher matrix J_t . This estimate is a gradient descent with preconditioning matrix J_t^{-1} , namely

$$J_t \leftarrow (1 - \gamma_t)J_{t-1} + \gamma_t \mathbb{E}_{y \sim p(y|\hat{y}_t)} \left[\frac{\partial \ell_t(y)}{\partial \theta}^{\otimes 2} \right] \quad (6)$$

$$\theta_t \leftarrow \theta_{t-1} - \eta_t J_t^{-1} \left(\frac{\partial \ell_t(y_t)}{\partial \theta} \right)^\top \quad (7)$$

with learning rate η_t and Fisher matrix decay rate γ_t .

In the Fisher matrix update, the expectation over all possible values $y \sim p(y|\hat{y})$ can often be computed algebraically, but this is sometimes computationally bothersome (for instance, in neural networks, it requires $\dim(\hat{y}_t)$ distinct backpropagation steps [Oll15]). A common solution [APF00, LMB07, Oll15, PB13] is to just use the value $y = y_t$ (*outer product* approximation) instead of the expectation over y . Another is to use a Monte Carlo approximation with a single sample of $y \sim p(y|\hat{y}_t)$ [Oll15, MCO16], namely, using the gradient of a synthetic sample instead of the actual observation y_t in the Fisher matrix. These latter two solutions are often confused; only the latter provides an unbiased estimate, see discussion in [Oll15, PB13].

The online “smoothed” update of the Fisher matrix in (6) mixes past and present estimates (the same or similar updates are used in [LMB07, MCO16]). The reason is at least twofold. First, the “genuine” Fisher matrix involves an expectation over the inputs u_t [AN00, §8.2]: this can be approximated online only via a moving average over inputs (e.g., $\gamma_t = 1/t$ realizes an equal-weight average over all inputs seen so far). Second, the expectation over $y \sim p(y|\hat{y}_t)$ in (6) is often replaced with a Monte Carlo estimation with only one value of y , and averaging over time compensates for this Monte Carlo sampling.

As a consequence, since θ_t changes over time, this means that the estimate J_t mixes values obtained at different values of θ , and converges to the

Fisher matrix only if θ_t changes slowly, i.e., if $\eta_t \rightarrow 0$. The correspondence below with Kalman filtering suggests using $\gamma_t = \eta_t$.

§2 Kalman filter viewpoint on this problem. The Kalman viewpoint on this statistical learning problem is that we are facing a system with hidden variable θ , with an unknown value that does not evolve in time, and that the observations y_t bring more and more information on θ .

One possible definition of the extended Kalman filter is as follows [Sim06, §15.1]. We are trying to estimate the current state of a dynamical system s_t whose evolution equation is known but whose precise value is unknown; at each time step, we have access to a noisy measurement y_t of a quantity $\hat{y}_t = h(s_t)$ which depends on this state. The Kalman filter maintains an approximation of a Bayesian posterior on s_t given the observations y_1, \dots, y_t , namely, the posterior distribution after t observations is approximated by a Gaussian with mean s_t and covariance matrix P_t . (Indeed, Bayesian posteriors always tend to Gaussians asymptotically under mild conditions, by the Bernstein–von Mises theorem.) The Kalman filter prescribes a way to update s_t and P_t when new observations become available.

The Kalman filter update is summarized in Definition 7 below. It is built to provide the *exact* value of the Bayesian posterior in the case of *linear* dynamical systems with Gaussian measurements and a Gaussian prior. In that sense, it is exact at first order.

Thus a statistical learning problem can be tackled by applying the extended Kalman filter to the unknown variable $s_t = \theta$ [SW88], whose underlying dynamics from time t to time $t+1$ is just to remain unchanged ($f = \text{Id}$ and noise on s is 0 in Definition 7). The initialization θ_0 and its covariance P_0 can be interpreted as Bayesian priors on θ [SW88, LS83].

For models with high-dimensional parameters such as neural networks, this is computationally costly and requires block-diagonal approximations for P_t (which is a square matrix of size $\dim \theta$); moreover, computing $H_t = \partial \hat{y}_t / \partial \theta$ is needed in the filter, and requires doing a separate backpropagation for each component of the output \hat{y}_t .

§3 Natural gradient as a Kalman filter: heuristics. As a first ingredient in the correspondence, we view Kalman filters as gradient descents: the extended Kalman filter actually performs a gradient descent on the likelihood of each new observation, with preconditioning matrix equal to the posterior covariance matrix. This is Proposition 8 below.

Meanwhile, the natural gradient uses the Fisher matrix as a preconditioning matrix. The Fisher matrix is the average Hessian of log-likelihood, thanks to the classical double definition of the Fisher matrix as square gradient or Hessian, $J(\theta) = \mathbb{E}_{y \sim p(y|\theta)} \left[\frac{\partial \ln p(y)}{\partial \theta} \otimes^2 \right] = -\mathbb{E}_{y \sim p(y|\theta)} \left[\frac{\partial^2 \ln p(y)}{\partial \theta^2} \right]$ for any probabilistic model $p(y|\theta)$ [Kul97].

Assume that the probability of the data given the parameter θ is approximately Gaussian, $p(y_1, \dots, y_t | \theta) \propto \exp(-(\theta - \theta^*)^\top \Sigma^{-1}(\theta - \theta^*))$ with covariance Σ (this often holds asymptotically thanks to the Bernstein–von Mises theorem; moreover, the posterior covariance Σ typically decreases like $1/t$). Then the Hessian (w.r.t. θ) of the total log-likelihood of (y_1, \dots, y_t) is Σ^{-1} , the inverse covariance of θ . So the *average* Hessian per data point, the Fisher matrix, is approximately Σ^{-1}/t . Therefore, a Kalman filter to estimate θ , which is essentially a gradient descent preconditioned with Σ^{-1} , will be the same as using a natural gradient with learning rate $1/t$.

Another way to understand the link between natural gradient and Kalman filter is as a second-order Taylor expansion of data log-likelihood. Assume that the total data log-likelihood at time t , $L_t(\theta) := -\sum_{s=1}^t \ln p(y_s | \theta)$, is approximately quadratic as a function of θ , with a minimum at θ_t^* and a Hessian h_t , namely, $L_t(\theta) \approx \frac{1}{2}(\theta - \theta_t^*)^\top h_t(\theta - \theta_t^*)$. Then when new data points become available, this quadratic approximation would be updated as follows (online Newton method):

$$h_t \approx h_{t-1} + \partial_\theta^2(-\ln p(y_t | \theta_{t-1}^*)) \quad (8)$$

$$\theta_t^* \approx \theta_{t-1}^* - h_t^{-1} \partial_\theta(-\ln p(y_t | \theta_{t-1}^*)) \quad (9)$$

and indeed these are *equalities* for a quadratic log-likelihood. Namely, the update of θ_t^* is a gradient ascent on log-likelihood, preconditioned by the inverse Hessian (Newton method). Note that h_t grows like t (each data point adds its own contribution). Thus, h_t is t times the empirical average of the Hessian, i.e., approximately t times the Fisher matrix of the model ($h_t \approx tJ$). So this update is approximately a natural gradient descent with learning rate $1/t$.

Meanwhile, the Bayesian posterior on θ (with uniform prior) after observations y_1, \dots, y_t is proportional to e^{-L_t} by definition of L_t . If $L_t \approx \frac{1}{2}(\theta - \theta_t^*)^\top h_t(\theta - \theta_t^*)$, this is a Gaussian distribution centered at θ_t^* with covariance matrix h_t^{-1} . The Kalman filter is built to maintain an approximation P_t of this covariance matrix.

These heuristics justify the statement from [LS83] that “there is only one recursive identification method”. Close to an optimum (so that the Hessian is positive), all second-order algorithms are essentially an online Newton step (8)-(9) approximated in various ways.

However, the correspondence between online natural gradient and Kalman filter presented below is exact, not approximate.

§4 Statement of the correspondence. For the statement of the correspondence, we assume that the output noise on y given \hat{y} is modelled by an exponential family with mean parameter \hat{y} . This covers the traditional Gaussian case $y = \mathcal{N}(\hat{y}, \Sigma)$ with fixed Σ often used in Kalman filters. The Appendix contains necessary background on exponential families.

THEOREM 2 (NATURAL GRADIENT AS KALMAN FILTER WITH CONSTANT DYNAMICS). *These two algorithms are identical under the correspondence $(\theta_t, J_t) \leftrightarrow (s_t, P_t^{-1}/(t+1))$:*

1. *The online natural gradient (Def. 1) with learning rates $\eta_t = \gamma_t = 1/(t+1)$, applied to learn the parameter θ of a model that predicts observations (y_t) with inputs (u_t) , using a probabilistic model $y \sim p(y|\hat{y})$ with $\hat{y} = h(\theta, u)$, where h is any model and $p(y|\hat{y})$ is an exponential family with mean parameter \hat{y} .*
2. *The extended Kalman filter (Def. 7) to estimate the state s from observations (y_t) and inputs (u_t) , using a probabilistic model $y \sim p(y|\hat{y})$ with $\hat{y} = h(s, u)$ and $p(y|\hat{y})$ an exponential family with mean parameter \hat{y} , with constant dynamics and no added noise on s ($f(s, u) = s$ and $Q = 0$ in Def. 7).*

Namely, if at startup $(\theta_0, J_0) = (s_0, P_0^{-1})$, then $(\theta_t, J_t) = (s_t, P_t^{-1}/(t+1))$ for all $t \geq 0$.

The correspondence is exact only if the Fisher metric is updated before the parameter in the natural gradient descent (as in Definition 1).

§5 Learning rates, priors, and metric decay rate. The correspondence with a Kalman filter provides an interpretation for various hyperparameters of online natural gradient descent, such as the learning rate η_t , the metric decay rate γ_t , and the initialization J_0 of the Fisher matrix.

In particular, $J_0 = P_0^{-1}$ can be interpreted as the inverse covariance of a Bayesian prior on θ [SW88]. This relates the initialization J_0 of the Fisher matrix to the initialization of θ : for instance, in neural networks it is recommended to initialize the weights according to a Gaussian of covariance $\text{diag}(1/\text{fan-in})$ (number of incoming weights) for each neuron; interpreting this as a Bayesian prior on weights, one may recommend to initialize the Fisher matrix to the inverse of this covariance, namely, $J_0 \leftarrow \text{diag}(\text{fan-in})$. Indeed this seemed to perform quite well in small-scale experiments.

Theorem 2 exhibits a $1/(t+1)$ learning rate for the online natural gradient. This is because the Kalman filter approximates the maximum a posteriori (MAP) of the parameter θ , and MAP and maximum likelihood estimators move by $O(1/t)$ when a new data point is observed. However, this interpretation is only valid if one assumes that the previous value θ_{t-1} was already close to the optimum for data up to $t-1$. In general, the argument for optimality of the $1/(t+1)$ learning rate breaks down if optimization does not start close to the optimum or if one is not using the exact Fisher matrix J_t or covariance matrix P_t .

Larger effective learning rates are achieved thanks to so-called “fading memory” variants of the Kalman filter, which put less weight on older observations. For instance, one may multiply the log-likelihood of previous

points by a forgetting factor $(1 - \lambda_t)$ before each new observation. This is equivalent to an additional step $P_t \leftarrow P_t/(1 - \lambda_t)$ in the Kalman filter, or to the addition of an artificial process noise Q_t proportional to P_t in the model. Such strategies are reported to often improve performance, especially when the data do not truly follow the model [Sim06, §5.5, §7.4], [Hay01, §5.2.2]. See for instance [Ber96] for the relationship between Kalman fading memory and gradient descent learning rates.

PROPOSITION 3 (NATURAL GRADIENT RATES AND FADING MEMORY). *Under the same model and assumptions as in Theorem 2, the following two algorithms are identical via the correspondence $(\theta_t, J_t) \leftrightarrow (s_t, \eta_t P_t^{-1})$:*

- An online natural gradient step with learning rate η_t and metric decay rate γ_t
- A fading memory Kalman filter that iteratively optimizes a weighted log-likelihood function L_t of recent observations, with decay $(1 - \lambda_t)$ at each step:

$$L_t(\theta) = \ln p_\theta(y_t) + (1 - \lambda_t) L_{t-1}(\theta), \quad L_0(\theta) := -\frac{1}{2}(\theta - \theta_0)^\top P_0^{-1}(\theta - \theta_0) \quad (10)$$

provided the following relations are satisfied:

$$\eta_t = \gamma_t, \quad J_0 = P_0^{-1}, \quad (11)$$

$$1 - \lambda_t = \eta_{t-1}/\eta_t - \eta_{t-1} \quad \text{for } t \geq 1, \text{ with } \eta_0 := 1 \quad (12)$$

For example, taking $\eta_t = 1/(t + \text{cst})$ corresponds to $\lambda_t = 0$, no decay for older observations. Equivalently, one can start with the decay factors λ_t and obtain the learning rates η_t via the cumulated sum of weights: $S_t = (1 - \lambda_t)S_{t-1} + 1$ (with any S_0), then $\eta_t = 1/S_t$ for $t \geq 1$.

This result suggests to set $\gamma_t = \eta_t$ in the online natural gradient descent of Definition 1. Still, one should keep in mind that the extended Kalman filter is itself only an approximation for nonlinear systems. So in practice, considering γ_t and η_t as hyperparameters to be tuned independently may still be beneficial, though $\gamma_t = \eta_t$ seems a good place to start.

Quite naturally, a small initial learning rate η_1 means we are initially giving more strength to the Bayesian prior and to the starting point θ_0 . Indeed, η_1 determines $1 - \lambda_1 = 1/\eta_1 - 1$ (by definition $\eta_0 = 1$), which will be greater than 1 for small η_1 : this means that the prior log-likelihood L_0 will have a large coefficient in L_1 and in the subsequent quantities L_t . For instance, with $\eta_t = 1/(t + t_0)$ for $t \geq 1$, the prior weighs as much as t_0 observations.

As a downside, the Bayesian interpretation is partially lost for a fading memory Kalman filter, because the Bayesian prior is forgotten too quickly.

For instance, with $\eta_t = O(1/\sqrt{t})$, the filter essentially works with the $O(\sqrt{t})$ most recent observations, while the weight of the Bayesian prior decreases at a rate $\approx e^{-\sqrt{t}}$ (as does the weight of the earliest observations; this is the product $\prod(1 - \lambda_t)$). But precisely, when working with fewer data points one may wish the prior to play a greater role.

The Bayesian interpretation can be restored by explicitly optimizing a combination of the log-likelihood of recent points, and the log-likelihood of the prior, as in the next Proposition.

From the natural gradient viewpoint, this translates both as a regularization of the Fisher matrix (often useful in practice to numerically stabilize its inversion) and of the gradient step. With a Gaussian prior, regularization of the latter manifests as an additional step towards θ_{prior} known as weight decay or Tikhonov regularization [Bis06, §3.3, §5.5].

PROPOSITION 4 (BAYESIAN REGULARIZATION OF THE FISHER MATRIX). *Let $\pi = \mathcal{N}(\theta_{\text{prior}}, \Sigma_0)$ be a Gaussian prior on θ . Under the same model and assumptions as in Theorem 2, the following two algorithms are equivalent:*

- A modified fading memory Kalman filter that iteratively optimizes $L_t(\theta) + n_{\text{prior}} \ln \pi(\theta)$ where L_t is a weighted log-likelihood function of recent observations with decay $(1 - \lambda_t)$:

$$L_t(\theta) = \ln p_\theta(y_t) + (1 - \lambda_t) L_{t-1}(\theta), \quad L_0 := 0 \quad (13)$$

initialized with $P_0 = \frac{\eta_1}{1 + n_{\text{prior}}\eta_1} \Sigma_0$.

- A regularized online natural gradient step with learning rate η_t and metric decay rate γ_t , initialized with $J_0 = \Sigma_0^{-1}$,

$$\theta_t \leftarrow \theta_{t-1} - \eta_t \left(J_t + \eta_t n_{\text{prior}} \Sigma_0^{-1} \right)^{-1} \left(\frac{\partial \ell_t(y_t)}{\partial \theta}^\top + \lambda_t n_{\text{prior}} \Sigma_0^{-1} (\theta - \theta_{\text{prior}}) \right) \quad (14)$$

provided the following relations are satisfied:

$$\eta_t = \gamma_t, \quad 1 - \lambda_t = \eta_{t-1}/\eta_t - \eta_{t-1}, \quad \eta_0 := \eta_1 \quad (15)$$

Thus, the regularization terms are fully determined by the choice of learning rates η_t , a prior such as $\mathcal{N}(0, 1/\text{fan-in})$ (for neural networks), and a value of n_{prior} such as $n_{\text{prior}} = 1$ (the prior weighs as much as n_{prior} data points). This holds both for regularization of the Fisher matrix $J_t + \eta_t n_{\text{prior}} \Sigma_0^{-1}$, and for regularization of the parameter via the extra gradient step $\lambda_t n_{\text{prior}} \Sigma_0^{-1} (\theta - \theta_{\text{prior}})$.

The relative strength of regularization in the Fisher matrix decreases like η_t . In particular, a constant learning rate results in a constant regularization.

The added gradient step $\lambda_t n_{\text{prior}} \Sigma_0^{-1}(\theta - \theta_{\text{prior}})$ is modulated by λ_t which depends on η_t ; this extra term pulls towards the prior θ_{prior} . The Bayesian viewpoint guarantees that this extra term will not ultimately prevent convergence of the gradient descent (as the influence of the prior vanishes when the number of observations increases).

It is not clear how much these recommendations for natural gradient descent coming from its Bayesian interpretation are sensitive to using only an approximation of the Fisher matrix.

§6 Recurrent models. Let us now consider non-memoryless models, i.e., models defined by a recurrent equation

$$\hat{y}_t = \Phi(\hat{y}_{t-1}, \theta, u_t) \quad (16)$$

with u_t the observations at time t . To save notation, we include here in \hat{y}_t the whole state of the model, including both the part that contains the prediction about y_t and all internal variables (e.g., all internal and output layers of a recurrent neural network, not only the output layer). The state \hat{y}_t , or a part thereof, defines a loss function $\ell_t(y_t) := -\ln p(y_t|\hat{y}_t)$ for each observation y_t .

The current state \hat{y}_t can be seen as a function which depends on θ via the whole trajectory. The derivative of the current state with respect to θ can be computed inductively just by differentiating the recurrent equation (16) defining \hat{y}_t :

$$\frac{\partial \hat{y}_t}{\partial \theta} = \frac{\partial \Phi(\hat{y}_{t-1}, \theta, u_t)}{\partial \theta} + \frac{\partial \Phi(\hat{y}_{t-1}, \theta, u_t)}{\partial \hat{y}_{t-1}} \frac{\partial \hat{y}_{t-1}}{\partial \theta} \quad (17)$$

Real-time recurrent learning [Jac02] uses this equation to keep an estimate G_t of $\frac{\partial \hat{y}_t}{\partial \theta}$. RTRL then uses G_t to estimate the gradient of the loss function ℓ_t with respect to θ via the chain rule, $\partial \ell_t / \partial \theta = (\partial \ell_t / \partial \hat{y}_t)(\partial \hat{y}_t / \partial \theta) = (\partial \ell_t / \partial \hat{y}_t) G_t$.

DEFINITION 5 (REAL-TIME RECURRENT LEARNING). *Given a recurrent model $\hat{y}_t = \Phi(\hat{y}_{t-1}, \theta_{t-1}, u_t)$, real-time recurrent learning (RTRL) learns the parameter θ via*

$$G_t \leftarrow \frac{\partial \Phi}{\partial \theta_{t-1}} + \frac{\partial \Phi}{\partial \hat{y}_{t-1}} G_{t-1}, \quad G_0 := 0 \quad (18)$$

$$g_t \leftarrow \frac{\partial \ell_t(y_t)}{\partial \hat{y}_t} G_t \quad (19)$$

$$\theta_t \leftarrow \theta_{t-1} - \eta_t g_t^\top \quad (20)$$

Since θ changes at each step, the actual estimate G_t in RTRL is only an approximation of the gradient $\frac{\partial \hat{y}_t}{\partial \theta}$ at $\theta = \theta_t$, valid in the limit of small learning rates η_t .

In practice, RTRL has a high computational cost due to the necessary storage of G_t , a matrix of size $\dim \theta \times \dim \hat{y}$. For large-dimensional models, backpropagation through time is usually preferred, truncated to a certain length in the past [Jae02]; [OTC15, TO17] introduce a low-rank, unbiased approximation of G_t .

There are several ways in which a Kalman filter can be used to estimate θ for such models.

1. A first possibility is to view \hat{y}_t as a function of θ via the whole trajectory, and to apply a Kalman filter on θ . This would require, in principle, recomputing the whole trajectory from time 0 to time t using the new value of θ at each step, and using RTRL to compute $\partial \hat{y}_t / \partial \theta$, which is needed in the filter. In practice, the past trajectory is not updated, and truncated backpropagation through time is used to approximate the derivative $\partial \hat{y}_t / \partial \theta$ [Jae02, Hay01].
2. A second possibility is the *joint Kalman filter*, namely, a Kalman filter on the pair (θ, \hat{y}_t) [Hay01, §5], [Sim06, §13.4]. This does not require going back in time, as \hat{y}_t is a function of \hat{y}_{t-1} and θ . This is the version appearing in Theorem 6 below.
3. A third possibility is the *dual Kalman filter* [WN96]: a Kalman filter for θ given \hat{y} , and another one for \hat{y} given θ . This requires to explicitly couple the two Kalman filters by manually adding RTRL-like terms to account for the (linearized) dependency of \hat{y} on θ [Hay01, §5].

Intuitively, the joint Kalman filter maintains a covariance matrix on (θ, \hat{y}_t) , whose off-diagonal term is the covariance between \hat{y}_t and θ . This term captures how the current state would change if another value of the parameter had been used. The decomposition (28) in the theorem makes this intuition precise in relation to RTRL.

THEOREM 6 (KALMAN FILTER ON (θ, \hat{y}) AS RTRL+NATURAL GRADIENT+STATE CORRECTION). Consider a recurrent model $\hat{y}_t = \Phi(\hat{y}_{t-1}, \theta_{t-1}, u_t)$. Assume that the observations y_t are predicted with a probabilistic model $p(y|\hat{y}_t)$ that is an exponential family with mean parameter a subset of \hat{y}_t .

Given an estimate G_t of $\partial \hat{y}_t / \partial \theta$, and an observation y , denote

$$g_t(y) := \frac{\partial \ell_t(y)}{\partial \hat{y}_t} G_t \quad (21)$$

the corresponding estimate of $\partial \ell_t(y) / \partial \theta$.

Then these two algorithms are equivalent:

- The extended Kalman filter on the pair (θ, \hat{y}) , initialized with covariance matrix $P_0^{(\theta, \hat{y})} = \begin{pmatrix} P_0^\theta & 0 \\ 0 & 0 \end{pmatrix}$, and with no process noise ($Q = 0$).

- A natural gradient RTRL algorithm with learning rate $\eta_t = 1/(t+1)$, defined as follows. The state, RTRL gradient and Fisher matrix have a transition step

$$\hat{y}_t \leftarrow \Phi(\hat{y}_{t-1}, \theta_{t-1}, u_t) \quad (22)$$

$$G_t \leftarrow \frac{\partial \Phi}{\partial \theta_{t-1}} + \frac{\partial \Phi}{\partial \hat{y}_{t-1}} G_{t-1}, \quad G_0 := 0 \quad (23)$$

$$J_t \leftarrow (1 - \eta_t) J_{t-1} + \eta_t \mathbb{E}_{y \sim p(y|\hat{y}_t)} \left[g_t(y)^{\otimes 2} \right], \quad J_0 := (P_0^\theta)^{-1} \quad (24)$$

and after observing y_t , the state and parameter are updated as

$$\delta\theta \leftarrow J_t^{-1} g_t(y_t)^\top \quad (25)$$

$$\theta_t \leftarrow \theta_{t-1} - \eta_t \delta\theta \quad (26)$$

$$\hat{y}_t \leftarrow \hat{y}_t - \eta_t G_t \delta\theta \quad (27)$$

Moreover, at each time t , the covariance matrix of the extended Kalman filter over (θ, \hat{y}) satisfies

$$P_t^{(\theta, \hat{y})} = \eta_t \begin{pmatrix} J_t^{-1} & J_t^{-1} G_t^\top \\ G_t J_t^{-1} & G_t J_t^{-1} G_t^\top \end{pmatrix} \quad (28)$$

Again, the expectation in the Fisher matrix may be estimated by a Monte Carlo sample $y \sim p(y|\hat{y}_t)$, or by just using the current observation $y = y_t$, as discussed after Definition 1.

As before, learning rates η_t different from $1/(t+1)$ can be obtained by introducing a fading memory (i.e., process noise Q proportional to P) in the joint Kalman filter. We omit the statement for simplicity, but it is analogous to Propositions 3 and 4.

The algorithm above features a state update (27) together with the parameter update; this is not commonly used in online recurrent neural network algorithms. In small-scale experiments, we have not found any clear effect of this; besides, such state updates must be applied cautiously if the range of possible values for \hat{y} is somehow constrained.

In the result above, the Kalman filter is initialized with a covariance matrix in which every uncertainty comes from uncertainty on θ rather than the initial state \hat{y}_0 . This has the advantage of making the correspondence algebraically simple, but is not a fundamental restriction. If modelling an initial uncertainty on \hat{y}_0 is important, one can always apply the theorem by incorporating the initial condition as an additional component of the parameter θ , with its own variance; in this case, G_0 must be initialized to Id on the corresponding component of θ , namely

$$\theta^{+\text{init}} := (\theta, \hat{y}_0)^\top, \quad G_0 := \frac{\partial \hat{y}_0}{\partial \theta^{+\text{init}}} = (0, \text{Id}) \quad (29)$$

and then Theorem 6 can be applied to $\theta^{+\text{init}}$.

Actually this operation is often not needed at all: indeed, if the dynamical system is such that the initial condition is forgotten reasonably quickly, then the initial covariance of \hat{y}_0 decreases (terms W in the proof below) and the Kalman covariance tends to the type (28) above exponentially fast, even without using $\theta^{+\text{init}}$. This is the case, for instance, for any stable linear dynamical system, as a consequence of Lemmas 13-14, and more generally for any system with geometric memory in the sense that $\frac{\partial \hat{y}_t}{\partial \hat{y}_{t-1}}$ is contracting for a fixed parameter and a given input.

The literature contains updates similar to the above for G_t , but more complex [LS83, Hay01]; this is, first, because they are expressed over the variable $\text{Cov}(\hat{y}_t, \theta) = G_t J_t^{-1}$ instead of G_t alone, second, because we have initialized the uncertainty on \hat{y}_0 to 0, and, third, because in dual rather than joint filter approaches, higher-order terms depending on second derivatives of F are sometimes included. Interestingly, there is some debate in the literature about whether to add some second-order corrections to the joint Kalman filter (especially [LS83, §2.3.3], see discussion in [Hay01, §5.3.4]). The interpretation above in terms of a natural gradient on top of RTRL makes it clear which terms are neglected: in particular, G_t is not updated after the update of θ and \hat{y}_t , so that G_t contains a mixture of derivatives at different values of θ over time. These corrections would depend on second derivatives of F (as in [Hay01, §5, Appendix A]), thus approximating to some extent a second-order extended Kalman filter (EKF2, [Sim06, §13.3]).

In terms of computational cost, for recurrent neural networks (RNNs), RTRL alone is already as costly as the joint Kalman filter (RTRL+natural gradient). Indeed, RTRL requires $(\dim \theta)$ forward tangent propagations at each step, each of which costs $O(\dim \theta)$ for a standard RNN model [Jae02], thus for a total cost of $O((\dim \theta)^2)$ per time step. The Fisher matrix is of size $(\dim \theta)^2$; if a single Monte Carlo sample $y \sim p(y|\hat{y}_t)$ is used, then the Fisher matrix update is rank-one and costs $O((\dim \theta)^2)$; the update of the *inverse* Fisher matrix can be maintained at the same cost thanks to the Woodbury matrix identity (as done, e.g., in [LMB07]). Thus, if RTRL is computationally affordable, there is little point in not using the Fisher matrix on top.

§7 Proof of the correspondences. For the proof of Theorem 2, we will start with the interpretation of the Kalman filter as a gradient descent (Proposition 8).

We first recall the exact definition and the notation we use for the extended Kalman filter.

DEFINITION 7 (EXTENDED KALMAN FILTER). Consider a dynamical system with state s_t , inputs u_t and outputs y_t ,

$$s_t = f(s_{t-1}, u_t) + \mathcal{N}(0, Q_t), \quad \hat{y}_t = h(s_t, u_t), \quad y_t \sim p(y|\hat{y}_t) \quad (30)$$

where $p(\cdot|\hat{y})$ denotes an exponential family with mean parameter \hat{y} (e.g., $y = \mathcal{N}(\hat{y}, R)$ with fixed covariance matrix R).

The extended Kalman filter for this dynamical system estimates the current state s_t given observations y_1, \dots, y_t in a Bayesian fashion. At each time, the Bayesian posterior distribution of the state given y_1, \dots, y_t is approximated by a Gaussian $\mathcal{N}(s_t, P_t)$ so that s_t is the approximate maximum a posteriori, and P_t is the approximate posterior covariance matrix. (The prior is $\mathcal{N}(s_0, P_0)$ at time 0.) Each time a new observation y_t is available, these estimates are updated as follows.

The transition step (before observing y_t) is

$$s_{t|t-1} \leftarrow f(s_{t-1}, u_t) \quad (31)$$

$$F_{t-1} \leftarrow \left. \frac{\partial f}{\partial s} \right|_{(s_{t-1}, u_t)} \quad (32)$$

$$P_{t|t-1} \leftarrow F_{t-1} P_{t-1} F_{t-1}^\top + Q_t \quad (33)$$

$$\hat{y}_t \leftarrow h(s_{t|t-1}, u_t) \quad (34)$$

and the observation step after observing y_t is

$$E_t \leftarrow \text{sufficient statistics}(y_t) - \hat{y}_t \quad (35)$$

$$R_t \leftarrow \text{Cov}(\text{sufficient statistics}(y)|\hat{y}_t) \quad (36)$$

(these are just the error $E_t = y_t - \hat{y}_t$ and the covariance matrix $R_t = R$ for a Gaussian model $y = \mathcal{N}(\hat{y}, R)$ with known R)

$$H_t \leftarrow \left. \frac{\partial h}{\partial s} \right|_{(s_{t|t-1}, u_t)} \quad (37)$$

$$K_t \leftarrow P_{t|t-1} H_t^\top (H_t P_{t|t-1} H_t^\top + R_t)^{-1} \quad (38)$$

$$P_t \leftarrow (\text{Id} - K_t H_t) P_{t|t-1} \quad (39)$$

$$s_t \leftarrow s_{t|t-1} + K_t E_t \quad (40)$$

For non-Gaussian output noise, the definition of E_t and R_t above via the mean parameter \hat{y} of an exponential family, differs from the practice of modelling non-Gaussian noise via a nonlinear function applied to Gaussian noise. This allows for a straightforward treatment of various output models, such as discrete outputs or Gaussians with unknown variance. In the Gaussian case with known variance our definition is fully standard. ²

²Non-Gaussian output noise is often modelled in Kalman filtering via a continuous nonlinear function applied to a Gaussian noise [Sim06, 13.1]; this cannot easily represent discrete random variables. Moreover, since the filter linearizes the function around the 0 value of the noise [Sim06, 13.1], the noise is still implicitly Gaussian, though with a state-dependent variance.

The proof starts with the interpretation of the Kalman filter as a gradient descent preconditioned by P_t . Compare this result and Lemma 11 to [Hay01, (5.68)–(5.73)].

PROPOSITION 8 (KALMAN FILTER AS PRECONDITIONED GRADIENT DESCENT). *The update of the state s in a Kalman filter can be seen as an online gradient descent on data log-likelihood, with preconditioning matrix P_t . More precisely, denoting $\ell_t(y) := -\ln p(y|\hat{y}_t)$, the update (40) is equivalent to*

$$s_t = s_{t|t-1} - P_t \left(\frac{\partial \ell_t(y_t)}{\partial s_{t|t-1}} \right)^\top \quad (41)$$

where in the derivative, ℓ_t depends on $s_{t|t-1}$ via $\hat{y}_t = h(s_{t|t-1}, u_t)$.

LEMMA 9 (ERRORS AND GRADIENTS). *When the output model is an exponential family with mean parameter \hat{y}_t , the error E_t is related to the gradient of the log-likelihood of the observation y_t with respect to the prediction \hat{y}_t by*

$$E_t = R_t \left(\frac{\partial \ln p(y_t|\hat{y}_t)}{\partial \hat{y}_t} \right)^\top$$

PROOF OF THE LEMMA.

For a Gaussian $y_t = \mathcal{N}(\hat{y}_t, R)$, this is just a direct computation. For a general exponential family, consider the natural parameter β of the exponential family which defines the law of y , namely, $p(y|\beta) = \exp(\sum_i \beta_i T_i(y))/Z(\beta)$ with sufficient statistics T_i and normalizing constant Z . An elementary computation (Appendix, (78)) shows that

$$\frac{\partial \ln p(y|\beta)}{\partial \beta_i} = T_i(y) - \mathbb{E}T_i = T_i(y) - \hat{y}_i \quad (42)$$

by definition of the mean parameter \hat{y} . Thus,

$$E_t = T(y_t) - \hat{y}_t = \left(\frac{\partial \ln p(y_t|\beta)}{\partial \beta} \right)^\top \quad (43)$$

where the derivative is with respect to the natural parameter β . To express the derivative with respect to \hat{y} , we apply the chain rule

$$\frac{\partial \ln p(y_t|\beta)}{\partial \beta} = \frac{\partial \ln p(y_t|\hat{y})}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial \beta}$$

and use the fact that, for exponential families, the Jacobian matrix of the mean parameter $\frac{\partial \hat{y}}{\partial \beta}$ is equal to the covariance matrix R_t of the sufficient statistics (Appendix, (86) and (81)). \square

LEMMA 10. *The extended Kalman filter satisfies $K_t R_t = P_t H_t^\top$.*

PROOF OF THE LEMMA.

This relation is known, e.g., [Sim06, (6.34)]. Indeed, using the definition of K_t , we have $K_t R_t = K_t(R_t + H_t P_{t|t-1} H_t^\top) - K_t H_t P_{t|t-1} H_t^\top = P_{t|t-1} H_t^\top - K_t H_t P_{t|t-1} H_t^\top = (\text{Id} - K_t H_t) P_{t|t-1} H_t^\top = P_t H_t^\top$. \square

PROOF OF PROPOSITION 8.

By definition of the Kalman filter we have $s_t = s_{t|t-1} + K_t E_t$. By Lemma 9, $E_t = R_t \left(\frac{\partial \ell_t}{\partial \hat{y}_t} \right)^\top$. Thanks to Lemma 10 we find $s_t = s_{t|t-1} + K_t R_t \left(\frac{\partial \ell_t}{\partial \hat{y}_t} \right)^\top = s_{t|t-1} + P_t H_t^\top \left(\frac{\partial \ell_t}{\partial \hat{y}_t} \right)^\top = s_{t|t-1} + P_t \left(\frac{\partial \ell_t}{\partial \hat{y}_t} H_t \right)^\top$. But by the definition of H , H_t is $\frac{\partial \hat{y}_t}{\partial s_{t|t-1}}$ so that $\frac{\partial \ell_t}{\partial \hat{y}_t} H_t$ is $\frac{\partial \ell_t}{\partial s_{t|t-1}}$. \square

The first part of the next lemma is known as the information filter in the Kalman filter literature, and states that the observation step for P is additive when considered on P^{-1} [Sim06, §6.2]: after each observation, the Fisher information matrix of the latest observation is added to P^{-1} .

LEMMA 11 (INFORMATION FILTER). *The update (38)–(39) of P_t in the extended Kalman filter is equivalent to*

$$P_t^{-1} \leftarrow P_{t|t-1}^{-1} + H_t^\top R_t^{-1} H_t \quad (44)$$

(assuming $P_{t|t-1}$ and R_t are invertible).

In particular, for dynamical systems with constant dynamics ($f(s, u) = s$ and $Q_t = 0$), the whole extended Kalman filter (32)–(40) is equivalent to

$$P_t^{-1} \leftarrow P_{t-1}^{-1} + H_t^\top R_t^{-1} H_t \quad (45)$$

$$s_t \leftarrow s_{t-1} - P_t \left(\frac{\partial \ell_t(y_t)}{\partial s_{t-1}} \right)^\top \quad (46)$$

PROOF.

The first statement is well-known for Kalman filters [Sim06, (6.33)]. Indeed, expanding the definition of K_t in the update (39) of P_t , we have

$$P_t = P_{t|t-1} - P_{t|t-1} H_t^\top \left(H_t P_{t|t-1} H_t^\top + R_t \right)^{-1} H_t P_{t|t-1} \quad (47)$$

but this is equal to $(P_{t|t-1}^{-1} + H_t^\top R_t^{-1} H_t)^{-1}$ thanks to the Woodbury matrix identity.

The second statement follows from Proposition 8 and the fact that for $f(s, u) = s$, the transition step of the Kalman filter is just $s_{t|t-1} = s_{t-1}$ and $P_{t|t-1} = P_{t-1}$. \square

LEMMA 12. For exponential families $p(y|\hat{y})$, the term $H_t^\top R_t^{-1} H_t$ appearing in Lemma 11 is equal to the Fisher information matrix of y with respect to the state s ,

$$H_t^\top R_t^{-1} H_t = \mathbb{E}_{y \sim p(y|\hat{y}_t)} \left[\frac{\partial \ell_t(y)}{\partial s_{t|t-1}}^{\otimes 2} \right]$$

where $\ell_t(y) = -\ln p(y|\hat{y}_t)$ depends on s via $\hat{y} = h(s, u)$.

PROOF.

Let us omit time indices for brevity. We have $\frac{\partial \ell(y)}{\partial s} = \frac{\partial \ell(y)}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial s} = \frac{\partial \ell(y)}{\partial \hat{y}} H$.

Consequently, $\mathbb{E}_y \left[\frac{\partial \ell(y)}{\partial s}^{\otimes 2} \right] = H^\top \mathbb{E}_y \left[\frac{\partial \ell(y)}{\partial \hat{y}}^{\otimes 2} \right] H$. The middle term $\mathbb{E}_y \left[\frac{\partial \ell(y)}{\partial \hat{y}}^{\otimes 2} \right]$ is the Fisher matrix of the random variable y with respect to \hat{y} .

Now, for an exponential family $y \sim p(y|\hat{y})$ in mean parameterization \hat{y} , the Fisher matrix with respect to \hat{y} is equal to the inverse covariance matrix of the sufficient statistics of y (Appendix, (89)). \square

PROOF OF THEOREM 2.

By induction on t . By the combination of Lemmas 11 and 12, the update of the Kalman filter with constant dynamics ($s_{t|t-1} = s_{t-1}$) is

$$P_t^{-1} \leftarrow P_{t-1}^{-1} + \mathbb{E}_{y \sim p(y|\hat{y}_t)} \left[\frac{\partial \ell_t(y)}{\partial s_{t-1}}^{\otimes 2} \right] \quad (48)$$

$$s_t \leftarrow s_{t-1} - P_t \left(\frac{\partial \ell_t(y_t)}{\partial s_{t-1}} \right)^\top \quad (49)$$

Defining $J_t = P_t^{-1}/(t+1)$, this update is equivalent to

$$J_t \leftarrow \frac{t}{t+1} J_{t-1} + \frac{1}{t+1} \mathbb{E}_{y \sim p(y|\hat{y}_t)} \left[\frac{\partial \ell_t(y)}{\partial s_{t-1}}^{\otimes 2} \right]$$

$$s_t \leftarrow s_{t-1} - \frac{1}{t+1} J_t^{-1} \left(\frac{\partial \ell_t(y_t)}{\partial s_{t-1}} \right)^\top$$

Under the identification $s_{t-1} \leftrightarrow \theta_{t-1}$, this is the online natural gradient update with learning rate $\eta_t = 1/(t+1)$ and metric update rate $\gamma_t = 1/(t+1)$. \square

The proof of Proposition 3 is similar, with additional factors $(1 - \lambda_t)$. Proposition 4 is proved by applying a fading memory Kalman filter to a modified log-likelihood $\bar{L}_0 := n_0 \ln \pi(\theta)$, $\bar{L}_t := \ln p_\theta(y_t) + (1 - \lambda_t) \bar{L}_{t-1} + \lambda_t n_0 \ln \pi(\theta)$ so that the prior is kept constant in \bar{L}_t .

We now turn to the proof for the recurrent case, involving a joint Kalman filter on (θ, \hat{y}) . The key is to decompose the Kalman covariance matrix of

the pair (θ, \hat{y}) into three variables (51): the covariance of θ , the correlation between θ and \hat{y} , and the part of the covariance of \hat{y} that does not come from its correlation with θ .

Then we find that the correlation between θ and \hat{y} is exactly the gradient $G = \frac{\partial \hat{y}}{\partial \theta}$ maintained by RTRL (Corollary 15), and that the covariance of θ essentially follows an independent Kalman filter related to the observations via G , and thus is a natural gradient for the same reasons as in the non-recurrent case.

PROOF OF THEOREM 6.

In the recurrent case, we are applying an extended Kalman filter to the state $s = \begin{pmatrix} \theta \\ \hat{y} \end{pmatrix}$ with transition function $f = \begin{pmatrix} \text{Id} \\ \Phi \end{pmatrix}$. Let us decompose the covariance matrix P_t of this system as

$$P_t = \begin{pmatrix} P_t^\theta & (P_t^{\theta\hat{y}})^\top \\ P_t^{\theta\hat{y}} & P_t^{\hat{y}} \end{pmatrix} \quad (50)$$

From now on, for simplicity we omit the time indices when no ambiguity is present.

By the theory of Schur complement for positive-semidefinite matrices [BV04, Appendix A.5.5], letting P^+ be any generalized inverse of P^θ , we know that $P^{\hat{y}} - P^{\theta\hat{y}}P^+P^{\theta\hat{y}\top}$ is positive-semidefinite and that $P^{\theta\hat{y}}(\text{Id} - P^+P^\theta) = 0$. The latter rewrites as $P^{\theta\hat{y}} = P^{\theta\hat{y}}P^+P^\theta$. Let us set $W := P^{\hat{y}} - P^{\theta\hat{y}}P^+P^{\theta\hat{y}\top}$ and $G := P^{\theta\hat{y}}P^+$. Then $P^{\theta\hat{y}} = GP^\theta$ and $W = P^{\hat{y}} - GP^\theta G^\top$. Thus, at each time t we can decompose P_t as

$$P_t = \begin{pmatrix} P^\theta & (GP^\theta)^\top \\ GP^\theta & W + GP^\theta G^\top \end{pmatrix} \quad (51)$$

without loss of generality, where W is positive-semidefinite. This decomposition tells us which part of the covariance of the current state \hat{y} comes from the covariance of the parameter θ via the dynamics of the system.

First, we will show that if $W_0 = 0$, then $W_t = 0$ for all t , and that in this case G_t satisfies the RTRL equation.

LEMMA 13. *Consider the extended Kalman filter on the pair $s = (\theta, \hat{y})^\top$ with transition function $f = (\theta, \Phi(\hat{y}, \theta, u))^\top$ and no added noise ($Q_t = 0$). Then the Kalman transition step (33) on P , expressed in the decomposition (51), is equivalent to*

$$P^\theta \leftarrow P^\theta \quad (52)$$

$$W \leftarrow \frac{\partial \Phi}{\partial \hat{y}} W \frac{\partial \Phi^\top}{\partial \hat{y}} \quad (53)$$

$$G \leftarrow \frac{\partial \Phi}{\partial \theta} + \frac{\partial \Phi}{\partial \hat{y}} G \quad (54)$$

This equation for G is the RTRL update.

PROOF OF THE LEMMA.

This is a direct computation using the Kalman transition step (33) for P . Indeed, the decomposition (51) of P rewrites as

$$P_t = \begin{pmatrix} \text{Id} & 0 \\ G & \text{Id} \end{pmatrix} \begin{pmatrix} P^\theta & 0 \\ 0 & W \end{pmatrix} \begin{pmatrix} \text{Id} & G^\top \\ 0 & \text{Id} \end{pmatrix} \quad (55)$$

Now, the Kalman transition step (33) for P is $P_{t|t-1} = \frac{\partial f}{\partial s} P_{t-1} \frac{\partial f^\top}{\partial s}$ when $Q = 0$. So the update is equivalent to replacing $\begin{pmatrix} \text{Id} & 0 \\ G & \text{Id} \end{pmatrix}$ with $\frac{\partial f}{\partial s} \begin{pmatrix} \text{Id} & 0 \\ G & \text{Id} \end{pmatrix}$ on both sides in (55). Here we have $\frac{\partial f}{\partial s} = \begin{pmatrix} \text{Id} & 0 \\ \frac{\partial \Phi}{\partial \theta} & \frac{\partial \Phi}{\partial \hat{y}} \end{pmatrix}$. This yields the result. \square

LEMMA 14. *Consider the extended Kalman filter on the pair $s = (\theta, \hat{y})^\top$ with transition function $f = (\theta, \Phi(\hat{y}, \theta, u))^\top$. Then the observation update (38)–(39) of P_t , expressed in the variables P^θ , W , and G , is given by*

$$P^\theta \leftarrow P^\theta - P^\theta G^\top (W + R + G P^\theta G^\top)^{-1} G P^\theta \quad (56)$$

$$W \leftarrow W - W(W + R)^{-1} W \quad (57)$$

$$G \leftarrow (\text{Id} - R^{-1} W) G \quad (58)$$

in that order, where R is given by (36). Moreover, if P^θ or W are invertible then their respective updates are equivalent to

$$(P^\theta)^{-1} \leftarrow (P^\theta)^{-1} + G^\top (W + R)^{-1} G \quad (59)$$

and

$$W^{-1} \leftarrow W^{-1} + R^{-1} \quad (60)$$

Thus, the updates for W , (53) and (57), are just the updates of an extended Kalman filter on \hat{y} alone, with covariance matrix W and noise measurement R . The update for P^θ is identical to an extended Kalman filter on θ where measurements are made on \hat{y} , with \hat{y} seen as a function of θ with derivative $\partial \hat{y} / \partial \theta = G$, and where the measurement noise on \hat{y} is $R + W$ (the measurement noise on y plus the covariance of \hat{y}). Thus, these two lemmas relate the joint Kalman filter on (θ, \hat{y}) to the dual Kalman filter that filters separately θ given \hat{y} and \hat{y} given θ , together with an estimate of $\partial \hat{y} / \partial \theta$. As far as we could check, this decomposition is specific to a situation in which one component (the parameter) is supposed to have constant underlying dynamics $\theta_{t+1} = \theta_t$.

PROOF OF THE LEMMA.

In our case, the function h of the extended Kalman filter is the function that sends (θ, \hat{y}) to \hat{y} . In particular, $H_t = (0, \text{Id})$.

First, if P^θ and W are invertible, then the updates (56), (57) for P^θ and W follow from the updates (59), (60) on their inverses, thanks to the Woodbury matrix identity. Since working on the inverses is simpler, we shall prove only the latter. Since (56), (57) are continuous in P^θ and W , the non-invertible case follows by continuity.

Starting again with the decomposition of P_t as a product (55), the inverse of P_t is

$$P_t^{-1} = \begin{pmatrix} \text{Id} & G^\top \\ 0 & \text{Id} \end{pmatrix}^{-1} \begin{pmatrix} P^\theta & 0 \\ 0 & W \end{pmatrix}^{-1} \begin{pmatrix} \text{Id} & 0 \\ G & \text{Id} \end{pmatrix}^{-1} \quad (61)$$

$$= \begin{pmatrix} (P^\theta)^{-1} + G^\top W^{-1} G & -G^\top W^{-1} \\ -W^{-1} G & W^{-1} \end{pmatrix} \quad (62)$$

From Lemma 11, the Kalman observation update for P_t amounts to adding $H^\top R^{-1} H$ to P_t^{-1} . Here $H = (0, \text{Id})$ so that $H^\top R^{-1} H$ is $\begin{pmatrix} 0 & 0 \\ 0 & R^{-1} \end{pmatrix}$. So the update for P_t amounts to

$$P_t^{-1} \leftarrow \begin{pmatrix} (P^\theta)^{-1} + G^\top W^{-1} G & -G^\top W^{-1} \\ -W^{-1} G & W^{-1} + R^{-1} \end{pmatrix} \quad (63)$$

To interpret this as an update on P^θ , W and G , we have to introduce new variables \tilde{W} , \tilde{G} , and \tilde{P}^θ such that (63) takes the original form (62) in these new variables.

Introducing $\tilde{W}^{-1} := W^{-1} + R^{-1}$, the update rewrites as

$$P_t^{-1} \leftarrow \begin{pmatrix} (P^\theta)^{-1} + G^\top W^{-1} G & -G^\top (\text{Id} - R^{-1} \tilde{W}) \tilde{W}^{-1} \\ -\tilde{W}^{-1} (\text{Id} - \tilde{W} R^{-1}) G & \tilde{W}^{-1} \end{pmatrix} \quad (64)$$

Introducing $\tilde{G} := (\text{Id} - \tilde{W} R^{-1}) G$ and $(\tilde{P}^\theta)^{-1} := (P^\theta)^{-1} + G^\top W^{-1} G - \tilde{G}^\top \tilde{W}^{-1} \tilde{G}$, we get back the original form (62). This provides the updates \tilde{W} and \tilde{G} for W and G . We still have to find a more explicit expression for $(\tilde{P}^\theta)^{-1}$.

Since we have defined \tilde{W} and \tilde{G} by identifying (63) with the original form (62), we have $\tilde{W} \tilde{G} = W G$ by construction. Thus

$$(\tilde{P}^\theta)^{-1} = (P^\theta)^{-1} + G^\top W^{-1} G - \tilde{G}^\top \tilde{W}^{-1} \tilde{G} \quad (65)$$

$$= (P^\theta)^{-1} + G^\top W^{-1} G - \tilde{G}^\top \tilde{W}^{-1} \tilde{W} \tilde{W}^{-1} \tilde{G} \quad (66)$$

$$= (P^\theta)^{-1} + G^\top W^{-1} G - G^\top W^{-1} \tilde{W} W^{-1} G \quad (67)$$

Thanks to the identity $A^{-1} - A^{-1} B A^{-1} = (A + (B^{-1} - A^{-1})^{-1})^{-1}$ for any matrices A and B (this follows from the matrix inversion formula $(A +$

$C)^{-1} = A^{-1} - A^{-1}(A^{-1} + C^{-1})^{-1}A^{-1}$ applied to $C = (B^{-1} - A^{-1})^{-1}$, we find

$$W^{-1} - W^{-1}\tilde{W}W^{-1} = (W + (\tilde{W}^{-1} - W^{-1})^{-1})^{-1} \quad (68)$$

but by definition, $\tilde{W}^{-1} = W^{-1} + R^{-1}$ so that

$$W^{-1} - W^{-1}\tilde{W}W^{-1} = (W + R)^{-1} \quad (69)$$

thus

$$(\tilde{P}^\theta)^{-1} = (P^\theta)^{-1} + G^\top(W + R)^{-1}G \quad (70)$$

which concludes the proof of the lemma. \square

Putting the last two lemmas side by side in the case $W = 0$, we obtain a much simpler update.

COROLLARY 15. *Consider the extended Kalman filter on the pair $s = (\theta, \hat{y})^\top$ with transition function $f(s) = (\theta, \Phi(\hat{y}, \theta, u))^\top$ and no added noise ($Q_t = 0$). Decompose the covariance P of the state s as in (51) using P^θ , G , W . If $W = 0$ and P^θ is invertible then performing the Kalman transition update followed by the observation update is equivalent to*

$$G \leftarrow \frac{\partial \Phi}{\partial \theta} + \frac{\partial \Phi}{\partial \hat{y}}G \quad (71)$$

$$(P^\theta)^{-1} \leftarrow (P^\theta)^{-1} + G^\top R^{-1}G \quad (72)$$

$$W \leftarrow 0 \quad (73)$$

in that order.

From this, the end of the proof of Theorem 6 essentially proceeds as in the non-recurrent case. Since we initialize W to 0 in Theorem 6, we have $W = 0$ at all times. As before, for exponential families R^{-1} is equal to the Fisher matrix with respect to \hat{y}_t , namely, $R^{-1} = \mathbb{E}_{y \sim p(y|\hat{y})} \left[\frac{\partial \ell_t(y)}{\partial \hat{y}_t}^{\otimes 2} \right]$ (Appendix). Now, the term $\mathbb{E}_{y \sim p(y|\hat{y})} [g_t(y)^{\otimes 2}]$ in the Fisher matrix update (24) uses $g_t(y) = \frac{\partial \ell_t(y)}{\partial \hat{y}_t} G_t$ (21) to estimate the derivative of the loss $\ell_t(y)$ with respect to θ . So the term $G^\top R^{-1}G$ in (72) coincides with the Fisher matrix update term $\mathbb{E}_{y \sim p(y|\hat{y})} [g_t(y)^{\otimes 2}]$ in (24). (Compare Lemma 12.) So if we just define $J_t := \eta_t (P_t^\theta)^{-1}$ with $\eta_t = 1/(t+1)$, the additive update (72) on $(P^\theta)^{-1}$ translates as the online Fisher matrix update (24) on J_t .

Moreover, since the Kalman gradient is an ordinary gradient preconditioned with the covariance matrix P_t (Proposition 8), the update of the pair (θ, \hat{y}) is

$$\begin{pmatrix} \theta_t \\ \hat{y}_t \end{pmatrix} \leftarrow \begin{pmatrix} \theta_{t-1} \\ \hat{y}_t \end{pmatrix} - P_t \begin{pmatrix} 0 \\ \frac{\partial \ell_t}{\partial \hat{y}_t}^\top \end{pmatrix} \quad (74)$$

(indeed ℓ_t does not depend explicitly on θ in recurrent models, only via the current state \hat{y}_t). Given the decomposition $P_t = \begin{pmatrix} P^\theta & (GP^\theta)^\top \\ GP^\theta & GP^\theta G^\top \end{pmatrix}$, this translates as

$$\begin{pmatrix} \theta_t \\ \hat{y}_t \end{pmatrix} \leftarrow \begin{pmatrix} \theta_{t-1} \\ \hat{y}_t \end{pmatrix} - \begin{pmatrix} P^\theta \\ GP^\theta \end{pmatrix} \left(\frac{\partial \ell_t}{\partial \hat{y}_t} G \right)^\top \quad (75)$$

which is the update in Theorem 6. \square

Appendix: reminder on exponential families

An *exponential family of probability distributions* on a variable x (discrete or continuous), with *sufficient statistics* $T_1(x), \dots, T_K(x)$, is the following family of distributions, parameterized by $\beta \in \mathbb{R}^K$:

$$p_\beta(x) = \frac{1}{Z(\beta)} e^{\sum_k \beta_k T_k(x)} \lambda(dx) \quad (76)$$

where $Z(\beta)$ is a normalizing constant, and $\lambda(dx)$ is any reference measure on x , such as the Lebesgue measure or any discrete measure. The family is obtained by varying the parameter $\beta \in \mathbb{R}^K$, called the *natural* or *canonical* parameter. We will assume that the T_k are linearly independent as functions of x (and linearly independent from the constant function); this ensures that different values of β yield distinct distributions.

For instance, Bernoulli distributions are obtained with λ the uniform measure on $x \in \{0, 1\}$ and with a single sufficient statistic $T(0) = 0, T(1) = 1$. Gaussian distributions with a fixed variance are obtained with $\lambda(dx)$ the Gaussian distribution centered on 0, and $T(x) = x$.

Another, often convenient parameterization of the same family is the following: each value of β gives rise to an average value \bar{T} of the sufficient statistics,

$$\bar{T}_k := \mathbb{E}_{x \sim p_\beta} T_k(x) \quad (77)$$

For instance, for Gaussian distributions with fixed variance, this is the mean, and for a Bernoulli variable this is the probability to sample 1.

Exponential families satisfy the identities

$$\frac{\partial \ln p_\beta(x)}{\partial \beta_k} = T_k(x) - \bar{T}_k, \quad \frac{\partial \ln Z}{\partial \beta_k} = \bar{T}_k \quad (78)$$

by a simple computation [AN00, (2.33)].

These identities are useful to compute the Fisher matrix J_β with respect to the variable β , as follows [AN00, (3.59)]:

$$(J_\beta)_{ij} := \mathbb{E}_{x \sim p_\beta} \left[\frac{\partial \ln p_\beta(x)}{\partial \beta_i} \frac{\partial \ln p_\beta(x)}{\partial \beta_j} \right] \quad (79)$$

$$= \mathbb{E}_{x \sim p_\beta} [(T_i(x) - \bar{T}_i)(T_j(x) - \bar{T}_j)] \quad (80)$$

$$= \text{Cov}(T_i, T_j) \quad (81)$$

or more synthetically

$$J_\beta = \text{Cov}(T) \quad (82)$$

where the covariance is under the law p_β . That is, for exponential families the Fisher matrix is the covariance matrix of the sufficient statistics. In particular it can be estimated empirically, and is sometimes known algebraically.

In this work we need the Fisher matrix with respect to the mean parameter \bar{T} ,

$$(J_{\bar{T}})_{ij} = \mathbb{E}_{x \sim p_\beta} \left[\frac{\partial \ln p_\beta(x)}{\partial \bar{T}_i} \frac{\partial \ln p_\beta(x)}{\partial \bar{T}_j} \right] \quad (83)$$

By substituting $\frac{\partial \ln p(x)}{\partial \alpha} = \frac{\partial \ln p(x)}{\partial \beta} \frac{\partial \beta}{\partial \alpha}$, the Fisher matrices J_α and J_β with respect to parameterizations α and β are related to each other via

$$J_\alpha = \frac{\partial \beta^\top}{\partial \alpha} J_\beta \frac{\partial \beta}{\partial \alpha} \quad (84)$$

(consistently with the interpretation of the Fisher matrix as a Riemannian metric and the behavior of metrics under change of coordinates [GHL87, §2.3]). So we need to compute $\partial \bar{T} / \partial \beta$. Using the log-trick

$$\partial \mathbb{E}_{x \sim p} f(x) = \mathbb{E}_{x \sim p} [f(x) \partial \ln p(x)] \quad (85)$$

together with (78), we find

$$\frac{\partial \bar{T}_i}{\partial \beta_j} = \frac{\partial \mathbb{E} T_i(x)}{\partial \beta_j} = \mathbb{E} [T_i(x)(T_j(x) - \bar{T}_j)] = \mathbb{E} [(T_i(x) - \bar{T}_i)(T_j(x) - \bar{T}_j)] = (J_\beta)_{ij} \quad (86)$$

so that

$$\frac{\partial \bar{T}}{\partial \beta} = J_\beta \quad (87)$$

(see [AN00, (3.32)], where η denotes the mean parameter) and consequently

$$\frac{\partial \beta}{\partial \bar{T}} = J_\beta^{-1} \quad (88)$$

so that we find the Fisher matrix with respect to \bar{T} to be

$$J_{\bar{T}} = \frac{\partial \beta^\top}{\partial \bar{T}} J_\beta \frac{\partial \beta}{\partial \bar{T}} \quad (89)$$

$$= J_\beta^{-1} J_\beta J_\beta^{-1} \quad (90)$$

$$= J_\beta^{-1} = \text{Cov}(T)^{-1} \quad (91)$$

that is, the Fisher matrix with respect to \bar{T} is the inverse covariance matrix of the sufficient statistics.

This gives rise to a simple formula for the natural gradient of expectations with

respect to the mean parameters. Denoting $\tilde{\nabla}$ the natural gradient,

$$\tilde{\nabla}_{\bar{T}} \mathbb{E}f(x) := J_{\bar{T}}^{-1} \frac{\partial \mathbb{E}f(x)}{\partial \bar{T}}^{\top} \quad (92)$$

$$= J_{\bar{T}}^{-1} \frac{\partial \beta^{\top}}{\partial \bar{T}} \frac{\partial \mathbb{E}f(x)}{\partial \beta}^{\top} \quad (93)$$

$$= J_{\beta} J_{\beta}^{-1} \frac{\partial \mathbb{E}f(x)}{\partial \beta}^{\top} \quad (94)$$

$$= \frac{\partial \mathbb{E}f(x)}{\partial \beta}^{\top} \quad (95)$$

$$= \mathbb{E} \left[f(x) \frac{\partial \ln p_{\beta}(x)}{\partial \beta} \right] \quad (96)$$

$$= \mathbb{E} [f(x)(T(x) - \bar{T})] \quad (97)$$

$$= \text{Cov}(f, T) \quad (98)$$

which in particular, can be estimated empirically.

References

- [Ama98] Shun-ichi Amari. Natural gradient works efficiently in learning. *Neural Comput.*, 10:251–276, February 1998.
- [AN00] Shun-ichi Amari and Hiroshi Nagaoka. *Methods of information geometry*, volume 191 of *Translations of Mathematical Monographs*. American Mathematical Society, Providence, RI, 2000. Translated from the 1993 Japanese original by Daishi Harada.
- [APF00] Shun-ichi Amari, Hyeyoung Park, and Kenji Fukumizu. Adaptive method of realizing natural gradient learning for multilayer perceptrons. *Neural Computation*, 12(6):1399–1409, 2000.
- [Ber96] Dimitri P. Bertsekas. Incremental least squares methods and the extended Kalman filter. *SIAM Journal on Optimization*, 6(3):807–822, 1996.
- [Bis06] Christopher M. Bishop. *Pattern recognition and machine learning*. Springer, 2006.
- [BL03] Léon Bottou and Yann LeCun. Large scale online learning. In *NIPS*, volume 30, page 77, 2003.
- [BRD97] M. Boutayeb, H. Rafaralahy, and M. Darouach. Convergence analysis of the extended Kalman filter used as an observer for nonlinear deterministic discrete-time systems. *IEEE transactions on automatic control*, 42(4):581–586, 1997.

- [BV04] Stephen Boyd and Lieven Vandenberghe. *Convex optimization*. Cambridge University Press, 2004.
- [GHL87] S. Gallot, D. Hulin, and J. Lafontaine. *Riemannian geometry*. Universitext. Springer-Verlag, Berlin, 1987.
- [GS15] Roger B. Grosse and Ruslan Salakhutdinov. Scaling up natural gradient by sparsely factorizing the inverse Fisher matrix. In *ICML*, pages 2304–2313, 2015.
- [Hay01] Simon Haykin. *Kalman filtering and neural networks*. John Wiley & Sons, 2001.
- [Jae02] Herbert Jaeger. Tutorial on training recurrent neural networks, covering BPTT, RTRL, EKF and the “echo state network” approach. Technical Report 159, German National Research Center for Information Technology, 2002.
- [Kul97] Solomon Kullback. *Information theory and statistics*. Dover Publications Inc., Mineola, NY, 1997. Reprint of the second (1968) edition.
- [LCL⁺17] Yubo Li, Yongqiang Cheng, Xiang Li, Xiaoqiang Hua, and Yuliang Qin. Information geometric approach to recursive update in nonlinear filtering. *Entropy*, 19(2):54, 2017.
- [LMB07] Nicolas Le Roux, Pierre-Antoine Manzagol, and Yoshua Bengio. Topmoumoute online natural gradient algorithm. In *Advances in Neural Information Processing Systems 20, Proceedings of the Twenty-First Annual Conference on Neural Information Processing Systems, Vancouver, British Columbia, Canada, December 3-6, 2007*, pages 849–856, 2007.
- [LS83] Lennart Ljung and Torsten Söderström. *Theory and Practice of Recursive Identification*. MIT Press, 1983.
- [MCO16] Gaétan Marceau-Caron and Yann Ollivier. Practical Riemannian neural networks. arXiv preprint arXiv:1602.08007, 2016.
- [MG15] James Martens and Roger B. Grosse. Optimizing neural networks with Kronecker-factored approximate curvature. In *ICML*, pages 2408–2417, 2015.
- [Oll15] Yann Ollivier. Riemannian metrics for neural networks I: feedforward networks. *Information and Inference*, 4(2):108–153, 2015.
- [OTC15] Yann Ollivier, Corentin Tallec, and Guillaume Charpiat. Training recurrent networks online without backtracking. arXiv preprint arXiv:1507.07680, 2015.

- [Pat16] Vivak Patel. Kalman-based stochastic gradient method with stop condition and insensitivity to conditioning. *SIAM Journal on Optimization*, 26(4):2620–2648, 2016.
- [PB13] Razvan Pascanu and Yoshua Bengio. Natural gradient revisited. *CoRR*, abs/1301.3584, 2013.
- [Sim06] Dan Simon. *Optimal state estimation: Kalman, H_∞ , and nonlinear approaches*. John Wiley & Sons, 2006.
- [ŠKT01] Miroslav Šimandl, Jakub Královec, and Petr Tichavský. Filtering, predictive, and smoothing Cramér–Rao bounds for discrete-time nonlinear dynamic systems. *Automatica*, 37(11):1703–1716, 2001.
- [SW88] Sharad Singhal and Lance Wu. Training multilayer perceptrons with the extended Kalman algorithm. In *NIPS*, pages 133–140, 1988.
- [TO17] Corentin Tallec and Yann Ollivier. Unbiased online recurrent optimization. arXiv preprint arXiv:1702.05043, 2017.
- [WN96] Eric A. Wan and Alex T. Nelson. Dual Kalman filtering methods for nonlinear prediction, smoothing and estimation. In *NIPS*, pages 793–799, 1996.